Question1

Solution :

(a):

First, calculate the square of all values in the array A and store in list L. Then compute sums $L[i] + L[j]$ for all $1 \leq i < j \leq n$ and check if it exist in an AVL tree, if exist, the answer of the question is yes, if not, insert the sum and continue to compute the sums until $i + 1 = j = n$.

(b):

First, calculate the square of all values in the array A and store in list L. Then compute sums $L[i] + L[j]$ for all $1 \leq i < j \leq n$ and check if it exist in a HashMap, if already exist, the answer of the question is yes, if not, mark this index as exist and continue to compute the sums until $i + 1 = j = n$.

Proof :

Calculate the square and compute all the sums should be $O(n^2)$ and search and insert in an AVL tree in the worst case should be $O(logn)$ so the time complexity of question (a) in the worst case should be $O(n^2logn)$. Because of the time complexity of the HashMap search and create should be $O(1)$ so the time complexity of question (b) should be $O(n^2)$.